Topic : Vehicle Rental System

Group no : MLB_03.01_05

Campus : Malabe ~~/ Metro / Matara / Kandy / Kurunegala / Kandy / Jaffna~~

Submission Date:

We declare that this is our own work, and this Assignment does not incorporate without acknowledgment any material previously submitted by anyone else in SLIIT or any other university/Institute. And we declare that each one of us equally contributed to the completion of this Assignment.

Registration No	**Name**	**Contact Number**
IT21021916	P. D. R. L. Panditharathne	0776455297
IT21024368	P.V. Thushan	0772940951
IT21028878	D.A.M.N. Kulasekara	0773318219
IT21032288	D.D.H.I Saparamadu	0705947815
IT21034404	P.G.I. Nipun	0704003162

1st year / 2nd semester 2021 / July

Object Oriented Concepts– IT1050 – Assignment 2

B.Sc. (Hons) in Information Technology

Exercise 1

1) **List down the requirements you have identified, for the system you need to implement. There should be at least 10 different requirements excluding the user login.**

In this online vehicle rental system, Guest (unregistered customer), Customer (registered customer), Driver, System Administrator are the three types of users.

1. Guest can search and view vehicles.
2. As a guest, he/she needs to register to the system, if not they are unable to rent vehicles.
3. The driver is an employee of the company; it is compulsory to register to the system if not driver cannot get his trip details.
4. The system allows customers and drivers to view their registration details and update them.
5. After registering the system, customers can make reservations for vehicles.
6. Customer can cancel reservations.
7. System validates the reservation and send a message to customer via system.
8. Customers can view their previous and current reservation details.
9. The system allows the driver to see his previous and upcoming trips.
10. The system calculates the charge for a trip and charges are displayed.
11. Customers need to make payment for their reservations via the system, using a debit card, credit card, or other payment cards.
12. System administrator allows to add new vehicles, update vehicle details, and delete vehicles.
13. Customers and drivers are managed by system administrator.
14. Administrator can make reports about rendered vehicles, Customer lists, and returned vehicles.

15. Customers can give their feedback via the system.

16. The system allows administrator to handle feedbacks.

17. All users (Guests, Customers, Drivers, Administrator) can view feedbacks.

2. **Do a Noun Verb analysis to the nouns you identify in your description. Show how the Nouns were rejected using the rules.**

RED – Nouns

BLUE – Verbs

1. Guest can search and view vehicles.
2. As a guest, he/she needs to register to the system, if not they are unable to rent vehicles.
3. The driver is an employee of the company; it is compulsory to register to the system if not driver cannot get his timetable.
4. The system allows customers and drivers to view their registration details and update them.
5. After registering the system, customers can make reservations for vehicles.
6. Customer can cancel reservations.
7. System validates the reservation and sends a message to customer via system.
8. Customers can view their previous and current reservation details.
9. The system allows the driver to see his previous and upcoming trips.
10. The system calculates the charge for a trip and charges are displayed.
11. Customers need to make payment for their reservations via the system, using a debit card, credit card, or other payment cards.
12. System administrator allows to add new vehicles, update vehicle details, and delete vehicles.

13. Customers and drivers are managed by system administrator.

14. Administrator can make reports about rented vehicles, Customer lists, and returned vehicles.

15. Customers can give their feedback via the system.

16. The system allows administrator to handle feedbacks.

17. All users (Guests, Customers, Drivers, Administrators) can view feedbacks.

Rejecting Nouns

Guest - class

Vehicles - class

System – Outside scope

Driver-Class

Employee-Redundant

Company-Outside scope of the system

Timetable-Class

Customers-Class

Registration details

Reservation details-Redundant

Message-Outside scope of the system

Trips-Redundant

Timetable-Redundant

Charge-Attribute

Reservations -Class

Payment-Class

Debit card-Redundant

Credit card-Redundant

Payment cards-Redundant

System administrators-Outside scope

Vehicle details-Redundant

Administrators-Outside scope

Reports-Class

Rented vehicles-Redundant

Customer lists-Redundant

Returned vehicles-Redundant

Feedback-Class

Users-Redundant

3. **List down the classes you have identified to implement your system using noun verb analysis**.

1) Guest
2) Vehicle
3) Driver
4) Customer
5) Reservation
6) Payment
7) Report
8) Feedback

Exercise 2

2) **Draw CRC cards for the classes you have identified in Exercise 1. Identify the Responsibilities and Collaborations of each class. You can further refine your classes by analyzing the requirements further.**

Class name: Guest	
Responsibilities	**Collaborations**
Register to the system	

Class name: Vehicle	
Responsibilities	**Collaborations**
Add vehicle	
Delete vehicle	
Update vehicle	
Display vehicle	

Class name: Driver	
Responsibilities	**Collaborations**
Register to the system	
View registration details	
Update registration details	

Class name: Customer	
Responsibilities	**Collaborations**
Login to the system	
View registration details	
Update registration details	

Class name: Reservation	
Responsibilities	**Collaborations**
Add reservation	
Validate reservations	Vehicle
Cancel reservation	

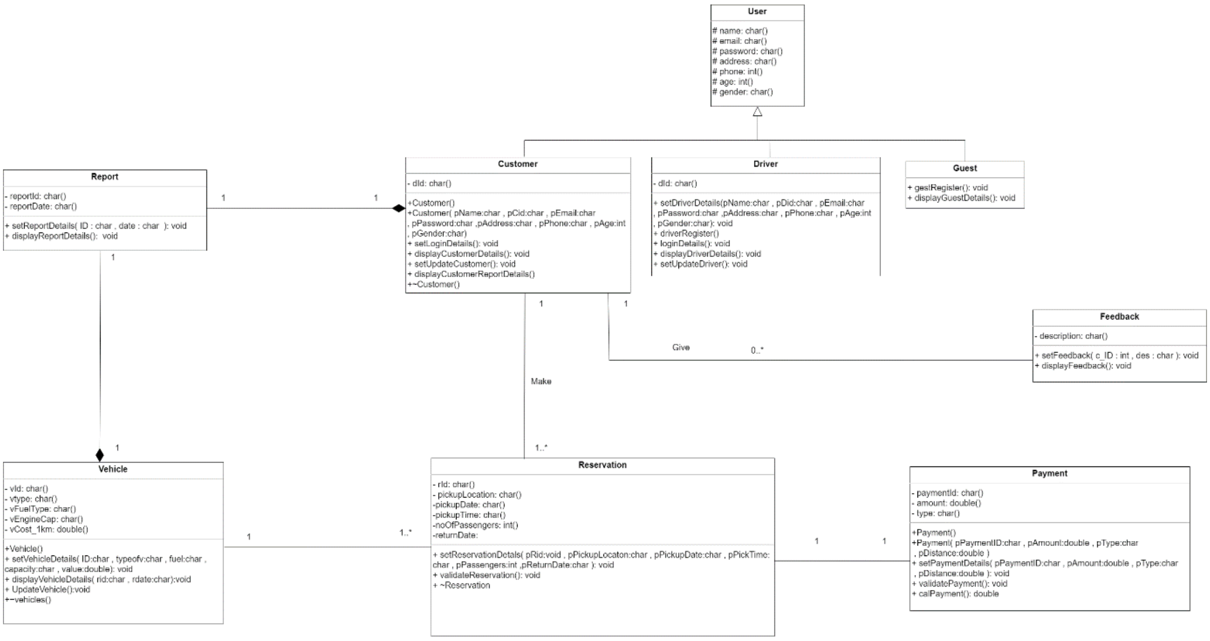
Class name: Payment	
Responsibilities	**Collaborations**
Get payment details	
Validate payment details	
Calculate payment	Reservation,Vehicle, Customer

Class name: Report	
Responsibilities	**Collaborations**
List of rented vehicles	Vehicle
List of Customers	Customer
List of Returned vehicles	Vehicle

Class name: Feedback	
Responsibilities	**Collaborations**
Store feedback	
View feedback	

Exercise 3

Draw the class diagram for the classes you have identified in your scenario. You need to use proper UML notations to draw the class diagram. Show the necessary relationships among classes using UML notations. Include the attributes and the methods in the classes.


User
# name: char()
# email: char()
# password: char()
# address: char()
# phone: int()
# age: int()
# gender: char()
Report
- reportId: char()
- reportDate: char()
+ setReportDetails(ID : char , date : char): void
+ displayReportDetails(): void
1
1
1
Customer
- dId: char()
+Customer()
+Customer(pName:char , pCid:char , pEmail:char
, pPassword:char ,pAddress:char , pPhone:char , pAge:int
, pGender:char)
+ setLoginDetails(): void
+ displayCustomerDetails(): void
+ setUpdateCustomer(): void
+ displayCustomerReportDetails()
+~Customer()
Driver
- dId: char()
+ setDriverDetails(pName:char , pDid:char , pEmail:char
, pPassword:char ,pAddress:char , pPhone:char , pAge:int
, pGender:char): void
+ driverRegister()
+ loginDetails(): void
+ displayDriverDetails(): void
+ setUpdateDriver(): void
Guest
+ gestRegister(): void
+ displayGuestDetails(): void
1
1
Give
0..*
Feedback
- description: char()
+ setFeedback(c_ID : int , des : char): void
+ displayFeedback(): void
Make
1
1..*
Vehicle
- vId: char()
- vtype: char()
- vFuelType: char()
- vEngineCap: char()
- vCost_1km: double()
+Vehicle()
+ setVehicleDetails(ID:char , typeofv:char , fuel:char ,
capacity:char , value:double): void
+ displayVehicleDetails(rid:char , rdate:char):void
+ UpdateVehicle():void
+~vehicles()
1
1..*
Reservation
- rId: char()
- pickupLocation: char()
-pickupDate: char()
-pickupTime: char()
-noOfPassengers: int()
-returnDate:
+ setReservationDetals(pRid:void , pPickupLocaton:char , pPickupDate:char , pPickTime:
char , pPassengers:int ,pReturnDate:char): void
+ validateReservation(): void
+ ~Reservation
1
1
Payment
- paymentId: char()
- amount: double()
- type: char()
+Payment()
+Payment(pPaymentID:char , pAmount:double , pType:char
, pDistance:double)
+ setPaymentDetails(pPaymentID:char , pAmount:double , pType:char
, pDistance:double): void
+ validatePayment(): void
+ calPayment(): double

Exercise 4

Write the coding for the classes in your class diagram. You can write the coding of the class and the constructors. You need not implement the methods. Write a main program to create the objects of the classes you have written.

```
//Customer class

#define SIZE 10
//Customer class

class Customer : public Guest{

    private :
      char cId[10];
      Report *creport;
      Reservation *r[SIZE];
      Feedback *f[SIZE];


    public :
      Customer();
      Customer( char const pName[],  char const pCid[] , char const pEmail[],
char const pPassword[],  char const pAddress[], const char pPhone[], int pAge,
char const pGender[];
      void setLoginDetails( );
      void displayCustomerDetails();
      void updateCustomer();
      void displayCustomerReportDetails();
      ~Customer();
};


 //Driver class


class Driver : public User{

  private :

    char dId[10];

  public :

     Driver ();
     void setDriverDetails( const char pName[] , const char pDid[] , const
char pEmail[] ,const char pPassword[] , const char pAddress[] , const char
pPhone[] , int pAge , const char pGender[] );
     void driverRegister();
     void logindetails( );
     void displayDriverDetails();
     void updateDriverDetails();

};
//Feedback class
```

```
class Feedback
{
  private:
  int feedbackID;
  char description[50];
  Customer*Cus;

  public:

  Feedback();
  void setFeedback(int c_ID, const char des[] ,Customer*pCus);
  void displayFeedback();

};

//Guest class

class Guest : public User{

  public:

    Guest();
    void guestRegister();
    void displayGuestDetails();
};


//Payment class

class Payment{
  private :
    char paymentID[10];
    double amount;
    char type[10];
    double distance;
    Reservation *res ;

  public :
    Payment();
    Payment(char const pPaymentID[],double pAmount, char const pType[], double
pDistance);
    void setPaymentDetails(char const pPaymentID[],double pAmount, char const
pType[], double pDistance);
    void validatePayment();
    double calPayment();
};
//Report class
```

```
class Report{
  private:
    char reportid[10];
    char reportDate[20];

  public:
    Report();
    void setReportDetails(char const ID[], char const date[]);
    void displayReportDetails();
    //void displayRenteredVehicles();
    //void displayReturnedVehices();
    //void displayCustomerReport();
};

class Reservation
{
    private :

       char rId[10];
       char pickupLocation[30];
       char pickupDate[15];
       char pickupTime[15];
       int noOfPassengers;
       char returnDate[15];
       Vehicle *v;
       payment *p;

    public :

       Reservation(const char pRid[] , const char pPickupLocation[] , const
char pPickupDate[] , const char pPickupTime[] , int pPassengers , const char
pReturnDate[] , Vehicle *pv );

       void displayReservation();

};

//Parent class
class User{

  protected :
    char name[30];
    char email[70];
    char password[10];
    char address[70];
    char phone[15];
    int age;
    char gender[10];
};

//Vehicle class

class Vehicle{

  private:
    char vid[10];
    char vtype[20];
    char vFuelType[10];
    char vEngineCap[10];
    double vCost_1km;
    Report *vreport;
    Reservation *r[SIZE];


  public:
    Vehicle();
    void setVehicleDetails(char const ID[], char const typeofv[], char const
fuel[], char const capacity[], double value);
    void displayVehicleDetails();
    void UpdateVehicle();
    void displayVehicleReportDeatails(char const rid[],char const rdate[]);
    ~Vehicle();
};
```

Function Implementation

```
//Customer

#include "User.h"
#include "Guest.h"
#include "Report.h"
#include "Reservation.h"
#include "Customer.h"
#include <iostream>
#include <cstring>
using namespace std ;

//Customer implementation
    Customer::Customer()
    {
      strcpy(name,"");
      strcpy(email,"");
      strcpy(password,"");
      strcpy(address,"");
      strcpy(phone , "");
      age=0;
      strcpy(gender,"");
      creport= new Report();
      //F = new Feedback();


    }
    Customer::Customer( char const pName[],  char const pCid[] , char const
pEmail[],  char const pPassword[],  char const pAddress[], const char
pPhone[], int pAge,  char const pGender[])
    {
        strcpy(name,pName);
        strcpy( cId , pCid );
        strcpy(email,pEmail);
        strcpy(password,pPassword);
        strcpy(address,pAddress);
        strcpy(phone,pPhone);
        age= pAge;
        strcpy(gender,pGender);

    }
    void Customer::setLoginDetails( )
    {

        cout << "-------Customer Login-------" << endl ;

         cout << "Customer ID    : " ;
         cin >> cId ;
         cout << "Email          : " ;
         cin >> email ;
         cout << "Password       : " ;
        cin >> password ;
    }
    void Customer::displayCustomerDetails()
    {
        cout << "-------Display Customer Details-------" << endl ;
        cout << "Name of the customer is     : "<<  name << endl;
        cout << "ID of the customer is : "<< cId << endl;
        cout << "Email of the customer is    : "<<  email << endl;
        cout << "Password of the customer is : "<<  password << endl;
        cout << "Address of the customer is  : "<<  address << endl;
        cout << "Phone of the customer is    : "<<  phone << endl;
        cout << "Age of the customer is      : "<<  age << endl;
        cout << "Gender of the customer is   : "<< gender << endl;

    }
    void Customer::updateCustomer()
    {
        cout << "-------Update Details-------" << endl ;

        cout << "Enter customer new Name :";
        cin >> name;
        cout << "Enter customer new Email :";
        cin >> email;
        cout << "Enter customer new Password :";
        cin >> password;
        cout << "Enter customer new Address :";
        cin >> address;
        cout << "Enter customer new Phone :";
        cin >> phone;
        cout << "Enter customer new Age :";
        cin >> age;
    }
void Customer::displayCustomerReportDetails()
{
        cout<<"------Customer Report Details--------"<<endl;
        creport ->setReportDetails("CR001","10/14/2021");
        creport ->displayReportDetails();
        displayCustomerDetails();
}
    Customer::~Customer()
    {
        cout << "Customer object deleted"<< endl;

          delete creport;

        cout << "everything deleted" << endl;
    }


//Driver

#include "User.h"
#include "Driver.h"
#include <iostream>
#include <string.h>
using namespace std ;

Driver::Driver ()
{
    cout << "Driver constructor is activated" << endl ;
}
void Driver::setDriverDetails( const char pName[] , const char pDid[] , const
char pEmail[] ,const char pPassword[] , const char pAddress[] , const char
pPhone[] , int pAge , const char pGender[] )
{
      strcpy ( name , pName );
      strcpy ( dId , pDid );
      strcpy ( email , pEmail);
      strcpy ( password , pPassword );
      strcpy ( address , pAddress );
      age = pAge;
      strcpy ( phone  , pPhone );
      strcpy ( gender, pGender);

}
 void Driver::driverRegister()
 {
      cout << "-------Driver Registration-------" << endl ;

      cout << "Name         : " ;
      cin >> name ;
      cout << "Driver ID    : " ;
      cin >> dId ;
      cout << "Email        : " ;
      cin >> email ;
      cout << "Password     : " ;
      cin >> password ;
      cout << "Address     : " ;
      cin >> address ;
      cout << "Age          : " ;
      cin >> age ;
      cout << "Phone number : " ;
      cin >> phone ;
      cout << "Gender       : " ;
      cin >> gender ;
 }
void Driver::logindetails(  )
{
      cout << "-------Driver Login-------" << endl ;

      cout << "Driver ID    : " ;
      cin >> dId ;
      cout << "Email        : " ;
      cin >> email ;
      cout << "Password     : " ;
      cin >> password ;

}
void Driver::displayDriverDetails()
{
     cout << "-------Display Driver Details-------" << endl ;

      cout << "Name         : " << name << endl ;
      cout << "Driver ID    : " << dId << endl ;
      cout << "Email        : " << email << endl ;
      cout << "Password     : " << password << endl ;
      cout << "Age          : " << age << endl ;
      cout << "Phone number : " << phone << endl ;
      cout << "Gender       : " << gender << endl ;
}
void Driver::updateDriverDetails()
{
      cout << "-------Update Details-------" << endl ;

      cout << "Name         : " ;
      cin >> name ;
      cout << "Driver ID    : " ;
      cin >> dId ;
      cout << "Email        : " ;
      cin >> email ;
      cout << "Password     : " ;
      cin >> password ;
      cout << "Age          : " ;
      cin >> age ;
      cout << "Phone number : " ;
      cin >> phone ;
      cout << "Gender       : " ;
      cin >> gender ;

      cout << endl;

     cout << "Your details are updated"  << endl ;

}


//Feedback

#include "Customer.h"
#include "Feedback.h"
#include <iostream>
#include <cstring>
using namespace std ;

void Feedback::setFeedback(int c_ID, const char des[]  ,Customer*pCus )
{
   feedbackID=c_ID;
   strcpy( description, des);
   Cus=pCus;

}
void Feedback::displayFeedback(){
  cout<<"Feedback ID   : "<<feedbackID<<endl;
  cout<<"Description   : "<<description<<endl;

}


#include "User.h"
#include <iostream>
#include <cstring>
#include "Guest.h"
using namespace std;

void Guest::guestRegister()
 {
      cout << "------- Guest Registration-------" << endl ;

      cout << "Name         : " ;
      cin >> name ;
      cout << "Email        : " ;
      cin >> email ;
      cout << "Password     : " ;
      cin >> password ;
      cout << "Address     : " ;
      cin >> address ;
      cout << "Phone number : " ;
      cin >> phone ;
      cout << "Age          : " ;
      cin >> age ;
      cout << "Gender       : " ;
      cin >> gender ;
 }

void Guest::displayGuestDetails()
{
     cout << "-------Display Guest Details-------" << endl ;

      cout << "Name         : " << name << endl;
      cout << "Email        : " << email << endl;
      cout << "Password     : " << password << endl;
      cout << "Address      : " << address << endl;
      cout << "Phone number : " << phone << endl;
      cout << "Age          : " << age << endl;
      cout << "Gender       : " << gender << endl;
}


#include "Reservation.h"
```

```
#include "Payment.h"
#include <iostream>
#include <cstring>
using namespace std ;

//Payment implementation

Payment::Payment()
{
  strcpy(paymentID,"");
  amount = 0;
  strcpy(type,"");
  distance = 0;
}
Payment::Payment(char const pPaymentID[],double pAmount, char const pType[],
double pDistance)
{
  strcpy(paymentID,pPaymentID);
  amount = pAmount;
  strcpy(type,pType);
  distance = pDistance;
}
void Payment::setPaymentDetails(char const pPaymentID[],double pAmount, char
const pType[], double pDistance)
{
  strcpy(paymentID,pPaymentID);
  amount = pAmount;
  strcpy(type,pType);
  distance = pDistance;
}
void Payment::validatePayment()
{
  int length;
  length = strlen(paymentID);
  if(length>1 && length<=10){
    cout << "Valid payment id"<< endl;
  }
  else {
    cout << "Sorry! can't validate details"<< endl;
  }

}
//1km-5km =>  1km=1500.00
//5km-10km =>  1km=1200.00
//20km>  =>  1km=1000.00
double Payment::calPayment()
{
  int ans;
  if(distance >20){
    ans = distance * 1000.00;
  }
  else if(distance<20 && distance>10){
    ans = distance * 1200.00;
  }
  else{
    ans = distance * 1500;
  }
  return ans;
}


//Report

#include <iostream>
#include "Report.h"
#include <cstring>

using namespace std;

Report::Report()
{
  cout<<"Activate the default constructor of Report class"<<endl;
}
void Report::setReportDetails(char const ID[], char const date[])
{
  strcpy(reportid,ID);
  strcpy(reportDate,date);
}
void Report::displayReportDetails()
{
    cout<<"Report ID : "<<reportid<<endl;
    cout<<"Report generated Date : "<<reportDate<<endl;
}

//Reservation

#include "Vehicle.h"
#include "Payment.h"
#include"Reservation.h"
#include <iostream>
#include <string.h>
using namespace std ;

Reservation::Reservation(const char pRid[] , const char pPickupLocation[] ,
const char pPickupDate[] , const char pPickupTime[] , int pPassengers , const
char pReturnDate[], Vehicle *pv )
```

```cpp
{
    strcpy(rId,pRid);
    strcpy(pickupLocation,pPickupLocation);
    strcpy ( pickupDate , pPickupDate );
    strcpy ( pickupTime , pPickupTime );
    noOfPassengers = pPassengers;
    strcpy ( returnDate , pReturnDate );
    v = pv ;


}
void Reservation::displayReservation()
{
   cout << "-------Reservation details-------" << endl ;

      cout << "Reservation ID         : " ;
      cin >> rId;
      cout << "PickupLocation         : " ;
      cin >> pickupLocation;
      cout << "PickupDate             : " ;
      cin >> pickupDate;
      cout << "PickupTime             : " ;
      cin >> pickupTime;
      cout << " Number of Passengers  : " ;
      cin >> noOfPassengers;
      cout << "ReturnDate             : " ;
      cin >> returnDate;

}

#include <iostream>
#include "Reservation.h"
#include "Report.h"
#include "Vehicle.h"
#define SIZE 10
#include <cstring>

using namespace std;

  Vehicle::Vehicle()
  {
     vreport= new Report();
  }
  void Vehicle::setVehicleDetails(char const ID[], char const typeofv[], char
const fuel[], char const capacity[], double value)
  {
      strcpy(vid,ID);
      strcpy(vtype,typeofv);
      strcpy(vFuelType,fuel);
      strcpy(vEngineCap,capacity);
      vCost_1km=value;
      Reservation *r[SIZE];
  }


  void Vehicle::displayVehicleDetails()
  {

      cout<<"Vehicle ID : "<<vid<<endl;
      cout<<"Vehicle Type: "<<vtype<<endl;
      cout<<"Fuel Type of the Vehicle : "<<vFuelType<<endl;
      cout<<"Engine Capacity of the Vehicle : "<<vEngineCap<<endl;
      cout<<"Cost for 1Km : "<<vCost_1km<<endl;

  }


  void Vehicle::UpdateVehicle()
  {
      cout<<"Enter the new Vehicle ID : ";
      cin>>vid;
      cout<<"Enter the new Vehicle Type : ";
      cin>>vtype;
      cout<<"Enter the new Fuel Type : ";
      cin>>vFuelType;
      cout<<"Enter the new Engine Capacity : ";
      cin>>vEngineCap;
      cout<<"Enter the new Cost for 1Km : ";
      cin>>vCost_1km;
  }

   void Vehicle::displayVehicleReportDeatails(char const rid[],char const
rdate[])
   {
      vreport->setReportDetails(rid,rdate);
      vreport ->displayReportDetails();
      displayVehicleDetails();
   }

   Vehicle::~Vehicle()
   {
     cout<<"destuctor activate"<<endl;
     delete vreport;
     cout<<"end"<<endl
 }
```

Main file

```
#include "User.h"
#include "Guest.h"
#include "Report.h"
#include "Customer.h"
#include "Driver.h"
#include "Feedback.h"
#include "Payment.h"
#include "Vehicle.h"
#include "Reservation.h"
#include <iostream>
using namespace std ;

int main() {

      Report *vr1;
      Vehicle *v1;

      v1= new Vehicle();
      vr1= new Report();

      v1 -> setVehicleDetails("VID001", "CAR", "PETROL", "1000CC", 120.00);
      cout<<"-------------Vehicle Details-------------"<<endl;
      cout<<endl;
      v1-> displayVehicleDetails();
      cout<<endl;
      cout<<"-------------Vehicle Report-------------"<<endl;
      cout<<endl;
      v1-> displayVehicleReportDeatails("RID001","10/14/2021");

    //====================================================

    //Guest
    Guest g1;

    //Gest registration
    g1.guestRegister();

    cout << endl;

    //Display guest details
    g1.displayGuestDetails();

    cout << endl;

    //====================================================
    //Customer

    //Customer c1;

    Customer c1( "Kamal" , "DID1234567" ,"kamal@gmail.com" , "1234" , "Kandy"
, "0773318219" , 34 , "Male" ) ;

   //display customer details
    c1.displayCustomerDetails();

    cout << endl;

    //Customer login
    c1.setLoginDetails( );

    cout << endl;

    //Customer is going to update details
    c1.updateCustomer();

    cout << endl;

    //display updated details
    cout << "*******Updated details*******" << endl;
    c1.displayCustomerDetails();

    cout << endl;

    //Display customer details
    c1.displayCustomerReportDetails();

    //Delete customer
    //c1.~Customer();

  //=======================================================

   Driver d1;

   cout << endl ;

  //Register to the system
  d1.driverRegister();

  cout << endl ;

  //Display d1 driver details
  d1.displayDriverDetails();
  cout << endl ;

  //Login to the system
  d1.logindetails( );

  cout << endl ;

  //driver is going to update phone number and password
  d1.updateDriverDetails();

  cout << endl ;

  cout << "*******Updated details*******" << endl;

  //display updated details
  d1.displayDriverDetails();

  delete v1;


  return 0;
}
```